IN THE CLAIMS

Please cancel claim 2 without prejudice.

1. (currently amended) A computer-implemented method for software error recovery, comprising:

compiling program source code into a first set of object code with a first compiler;

compiling the program source code into a second set of object code with a second compiler;

identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

associating sets of data objects with the checkpoints;

automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

executing the first set of object code;

storing the state information in executing the checkpoint code; ~~and~~

~~upon detecting an error in execution of the first set of object code, resuming execution of the program using the second set of object code.~~

upon detecting an error in execution of the first set of object code, initially re-executing the first set of object code; and

resuming execution using the second set of object code if the first set of object code fails in re-execution.

2. (canceled)

3. (currently amended) The method of claim 1 ~~2~~, further comprising re-executing the first set of object code a selected number of times before resuming execution using the second set of object code.

4. (original) The method of claim 3, further comprising ceasing resumption of execution of the first and second sets of object code if an error is detected in executing both sets of object code.

5. (original) A computer-implemented method for software error recovery, comprising:

compiling program source code into a first set of object code with a first compiler;

compiling the program source code into a second set of object code with a second compiler;

identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

associating sets of data objects with the checkpoints;

automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

executing the first set of object code;

storing the state information in executing the checkpoint code; and

upon detecting an error in execution of the first set of object code, selecting between the first set of object code and the second set of object code in resuming execution of the program.

6. (original) The method of claim 5, further comprising:

upon detecting an error in execution of the first set of object code, initially re-executing the first set of object code; and

resuming execution using the second set of object code if the first set of object code fails in re-execution.

7. (original) The method of claim 6, further comprising re-executing the first set of object code a selected number of times before resuming execution using the second set of object code.

8. (original) The method of claim 7, further comprising ceasing resumption of execution of the first and second sets of object code if an error is detected in executing both sets of object code.

9. (currently amended) An apparatus for software error recovery, comprising:

means for compiling program source code into a first set of object code with a first compiler;

means for compiling the program source code into a second set of object code with a second compiler;

means for identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

means for associating sets of data objects with the checkpoints; and

means for automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

means for executing the first set of object code;

means for storing the state information in executing the checkpoint code; ~~and~~

~~means for resuming execution of the program using the second set of object code upon detecting an error in execution of the first set of object code.~~

means, responsive to an error detected in execution of the first set of object code, for initially re-executing the first set of object code; and

means for resuming execution using the second set of object code if the first set of object code fails in re-execution.

10. (currently amended) An apparatus for software error recovery, comprising:

means for compiling program source code into a first set of object code with a first compiler;

means for compiling the program source code into a second set of object code with a second compiler;

means for identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

means for associating sets of data objects with the checkpoints;

means for automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

means for executing the first set of object code;

means for storing the state information in executing the checkpoint code; and

means for selecting between the first set of object code and the second set of object code in resuming execution of the program upon detecting an error in execution of the first set of object code.

11. (previously presented) A computer program product configured for causing a computer to perform the steps comprising:

compiling program source code into a first set of object code with a first compiler;

compiling the program source code into a second set of object code with a second compiler;

identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

associating sets of data objects with the checkpoints;

automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

executing the first set of object code;

storing the state information in executing the checkpoint code; and

upon detecting an error in execution of the first set of object code, selecting between the first set of object code and the second set of object code in resuming execution of the program.

12. (previously presented) The computer program product of claim 11, further configured for causing a computer to perform the steps comprising:

upon detecting an error in execution of the first set of object code, initially re-executing the first set of object code; and

resuming execution using the second set of object code if the first set of object code fails in re-execution.

13. (previously presented) The computer program product of claim 12, further configured for causing a computer to perform the step comprising re-executing the first set of object code a selected number of times before resuming execution using the second set of object code.

14. (previously presented) The computer program product of claim 13, further configured for causing a computer to perform the step comprising ceasing resumption of execution of the first and second sets of object code if an error is detected in executing both sets of object code.

15. (new) A processor-based method for software error recovery, comprising:

compiling program source code into a first set of object code with a first compiler;

compiling the program source code into a second set of object code with a second compiler, wherein functions implemented by the first and second code sets are identical, and code in the first set of object code is different from code in the second set of object code;

identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

associating sets of data objects with the checkpoints;

automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

executing the first set of object code and not executing the second set of object code while the first set of object code is executing;

storing the state information in executing the checkpoint code; and

in response to detecting an error in execution of the first set of object code, selecting one of the first set of object code and the second set of object code in resuming execution of the program, and not executing the non-selected one of the first and second sets of object code.

16. (new) The method of claim 15, further comprising:

in response to detecting an error in execution of the first set of object code, initially re-executing the first set of object code; and

resuming execution using the second set of object code in response to the first set of object code failing in re-execution.

17. (new) The method of claim 16, further comprising re-executing the first set of object code a selected number of times before resuming execution using the second set of object code.

18. (new) The method of claim 17, further comprising ceasing resumption of execution of the first and second sets of object code in response to detection of an error in executing both sets of object code.

19. (new) An apparatus for software error recovery, comprising:

means for compiling program source code into a first set of object code with a first compiler;

means for compiling the program source code into a second set of object code with a second compiler, wherein functions implemented by the first and second sets of object code are identical, and code in the first set of object code is different from code in the second set of object code;

means for identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

means for associating sets of data objects with the checkpoints;

means for automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

means for executing the first set of object code, wherein the second set of object code is not executed while the first set of object code is executing;

means for storing the state information in executing the checkpoint code; and

means, responsive to detection of an error in executing the first set of object code, for selecting between the first set of object code and the second set of object code in resuming execution of the program, wherein the non-selected one of the first and second sets of object code is not executed.

20. (new) A program storage device, comprising:

a processor-readable medium configured with processor-executable instructions for causing a processor to perform the steps including,

compiling program source code into a first set of object code with a first compiler;

compiling the program source code into a second set of object code with a second compiler, wherein functions implemented by the first and second sets of object code are identical, and code in the first set of object code set is different from code in the second set of object code;

identifying checkpoints in the first and second sets of object code, each checkpoint in the first set of object code corresponding to a checkpoint in the second set of object code;

associating sets of data objects with the checkpoints;

automatically generating executable checkpoint code for execution at the checkpoints, the checkpoint code configured to store state information of the associated data objects for recovery if execution of the program is interrupted;

executing the first set of object code and not executing the second set of object code while the first set of object code is executing;

storing the state information in executing the checkpoint code; and

in response to detecting an error in execution of the first set of object code, selecting between the first set of object code and the second set of object code in resuming execution of the program, and not executing the non-selected one of the first and second sets of object code.

21. (new) The program storage device of claim 20, wherein the processor-readable medium is further configured with instruction for causing a processor to perform the steps including,

in response to detecting an error in execution of the first set of object code, initially re-executing the first set of object code; and

resuming execution using the second set of object code in response to the first set of object code failing in re-execution.

22. (new) The program storage device of claim 21, wherein the processor-readable medium is further configured with instruction for causing a processor to perform the step including, re-executing the first set of object code a selected number of times before resuming execution using the second set of object code.

23. (new) The program storage device of claim 22, wherein the processor-readable medium is further configured with instruction for causing a processor to perform the steps including,

ceasing resumption of execution of the first and second sets of object code in response to detection of an error in executing both sets of object code.